AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions of claims in the application:

Listing of Claims:

1. (Previously Presented) A computer implemented system for generating specialized executables, comprising the following computer executable components:

a virtual subsystem that processes a generic code image;

a loader to determine availability of a specialized image that is associated with an operating environment of the virtual subsystem; and

a log to store generic code image runtime information relating to the operating environment of the virtual subsystem, the runtime logged information includes at least a set of information related to a particular user to create a native executable according to the particular user, the logged information is employed as feedback to generate the native executable based upon the availability of the specialized image, the native executable is utilized to provide improved performance of the virtual subsystem.

2. (Original) The system of claim 1, wherein the native executable is selected for execution by the virtual subsystem by matching a current environment setting with the logged information.

3. (Previously Presented) The system of claim 1, further comprising an image repository to store 1 through N specialized native images, wherein N is a positive integer.

4. (Original) The system of claim 3, wherein the image repository is a local or remote database.

5. (Original) The system of claim 1, further comprising at least one of a local or remote data log to store the logged information.

6. (Previously Presented) The system of claim 5, wherein a data log stores 1 though N environment parameter descriptions associated with 1 to N encountered images, wherein N is a positive integer.

7. (Original) The system of claim 1, wherein the virtual subsystem includes at least one of an execution engine and a virtual machine, and wherein the generic code image is an intermediate language image.

8. (Original) The system of claim 7, wherein the virtual subsystem employs Just-In-Time compilation on the generic code image.

9. (Previously Presented) The system of claim 7, wherein the virtual subsystem performs at least one of loads related executables, organizes data fields and methods within the generic image, analyzes intermediate language formats and generates native platform code.

10. (Previously Presented) The system of claim 7, wherein the virtual subsystem creates the native executable the time the generic code image is installed.

11. (Previously Presented) The system of claim 1, wherein the logged information includes a set of information to create executable images according to at least one of a method of invocation and a usage pattern.

12. (Previously Presented) The system of claim 1, wherein a native code image is generated by the virtual subsystem.

13. (Previously Presented) The system of claim 1, further comprising an image generator for processing the feedback and generating the native executable.

14. (Original) The system of claim 13, wherein the image generator further comprises at least one of a compiler and a code processor.

15. (Original) The system of claim 14, further comprising an image processing tool to read the logged information and associate one or more environmental settings with one or more related images encountered during virtual subsystem execution.

16. (Original) The system of claim 1, wherein the logged information includes parameters relating to at least one of an operating system version, a processor type, virtual system version, processor specifiers, developer parameters, domain flags, security information, binding information, administrative flags, and profile information associated with the operating environment of the virtual subsystem.

17. (Original) The system of claim 16, wherein the parameters are associated with an identifier to enable selection the native executable.

18. (Original) The system of claim 16, wherein the developer parameters describe at least one of debug options, compiler switch settings and information relating to preferences of a user.

19. (Original) A computer-readable medium having computer-executable components for executing the system of claim 1.

20. (Previously Presented) A computer implemented method to generate runtime code, comprising the following computer executable acts:

determining a first code image associated with a possible runtime environment;

executing the first code image in an unmodified form in the runtime environment; and

generating runtime feedback associated with the first code image and a particular user to adjust a subsequent code image according to the runtime environment, the feedback includes at least a set of information to create a code image according to the particular user.

21. (Original) The method of claim 20, further comprising,

generating a specialized executable from the subsequent code image.

22. (Original) The method of claim 21, further comprising,
storing the specialized executable in an image repository.

23. (Previously Presented) The method of claim 21, further comprising,
processing a generic intermediate language image utilizing standard compilation techniques.

24. (Original) The method of claim 23, further comprising,
logging operating environment information during processing of the generic image.

25. (Original) The method of claim 23, further comprising,
building the specialized executable from the environment information.

26. (Original) The method of claim 25, further comprising,
selecting the specialized executable by matching a current environment setting with the logged environment information.

27. (Previously Presented) The method of claim 20, further comprising at least one of:
loading executables with the first code image;
organizing data fields and methods within the first code image; and
analyzing intermediate language formats to generate native platform code.

28. (Previously Presented) A system to generate runtime code, comprising:
means for determining a specialized code image associated with a possible runtime environment;
means for executing the specialized code image in an unmodified form in the runtime environment; and
means for generating runtime feedback associated with the specialized code image and a particular user to adjust a subsequent specialized code image according to the runtime environment, the feedback includes at least a set of information to create a specialized code image according to the particular user.

29. (Original) The system of claim 28, further comprising,

means for generating a specialized executable from the subsequent code image.

30. (Currently Amended) A computer implemented signal transmitted between at least two ~~computer executable~~ processes executing on one or more computers facilitating generation of specialized executables, comprising:

a signal for communicating between one or more components of a virtual system, the virtual system processing a generic code image and logging information relating to an operating environment and a particular user of the virtual system *via* the signal, the logged information includes at least a set of information to create a specialized native executable according to the particular user;

wherein the logged information is employed as feedback across the signal to generate a specialized native executable if the generic code image is determined incompatible with the operating environment of the virtual system, the specialized native executable is utilized to provide improved performance of the virtual system.

31. (Original) The signal of claim 31, wherein the signal is communicated over at least one of a network system and a wireless system.

32. (Previously Presented) A computer implemented means having stored thereon a data structure employed to create an executable image, comprising:

a first data field having parameters relating to at least one of an operating system version, a processor type, a virtual system version, and a processor specifier;

a second data field having at least one of a developer parameter, a domain flag, a security information field, and a binding information field; and

a third data field having at least a set of information associated with a particular user that is logged during execution of a virtual system to create an executable image according to the particular user of the virtual system.

33. (Currently Amended) A computer implemented ~~virtual software~~ system, comprising the following computer executable components:

an execution engine that processes an Intermediate Language (IL) image, the execution engine generating operating environment data associated with a particular user while processing the IL image;

a specialized executable image generated at least in part from the operating environment data, the operating environment data includes at least a set of information to create a specialized executable image according to the particular user, the specialized executable image stored in a repository of one or more other specialized executable images; and

wherein the execution engine selects at least one specialized executable image from the repository if the at least one specialized image matches present operating environment data.